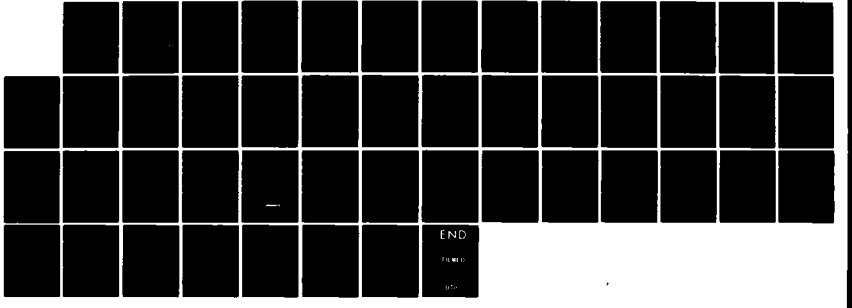
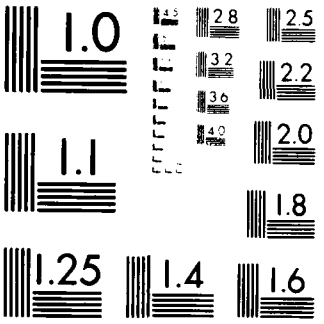
AD-A149 784 BUILT-IN TESTS FOR VLSI (VERY LARGE SCALE INTEGRATION) 1/1
FINITE-STATE MACHINES(U) ILLINOIS UNIV AT URBANA
COMPUTER SYSTEMS GROUP K A HUA AUG 84 CSG-33

UNCLASSIFIED F/G 9/2 NL





REPORT CSG-33 AUGUST 1984

(2)

BUILT-IN TESTS FOR VLSI FINITE-STATE MACHINES

KEN ANH HUA

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS none
2a. SECURITY CLASSIFICATION AUTHORITY N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CSG #33	5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Laboratory University of Illinois 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Semiconductor Research Corporation
6c. ADDRESS (City, State and ZIP Code) 1101 West Springfield Avenue Urbana, IL 61801	7b. ADDRESS (City, State and ZIP Code) 300 Park Drive, Suite 215 P.O. Box 12053 Research Triangle Park, NC 27709
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Semiconductor Research Corp. 8b. OFFICE SYMBOL (If applicable) N/A	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER SRC RSCH 83-01-014
8c. ADDRESS (City, State and ZIP Code) 300 Park Drive, Suite 215 P.O. Box 12053 Research Triangle Park, NC 27709	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO.: N/A PROJECT NO.: N/A TASK NO.: N/A WORK UNIT NO.: N/A
11. TITLE (Include Security Classification) Built-In Tests for VLSI Finite-State Machines	
12. PERSONAL AUTHOR(S) Hua, Kien Anh	
13a. TYPE OF REPORT Technical 13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (Yr., Mo., Day) August 1984 15. PAGE COUNT 44
16. SUPPLEMENTARY NOTATION N/A	
17. COSATI CODES FIELD / GROUP / SUB. GR.	18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number) built-in test, VLSI finite-state machines, programmable logic arrays, universal test set

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The introduction of LSI/VLSI technology has increased the difficulty of both designing and testing the complex systems which can be implemented on a chip. The testing problem, especially, is felt to be the primary bottleneck to the widespread use of the next generation of VLSI chips. Design for testability has been proposed as a solution to this problem.

We have developed a built-in test scheme for VLSI finite-state machines using PLAs. Unlike other built-in test schemes for PLA, the difficulty in interconnecting the test logic and the naked PLA is considered. The layout has been done for the test logic cells and example PLA in nMOS technology to demonstrate that all the test logic cells line up perfectly with the PLA cells. This eliminates the interconnection problems.

This scheme, using a universal test set, allows a finite-state machine implemented with a PLA to be tested within a number of cycles that is proportional to the number of inputs and product terms. By modifying the single-bit decoders, a test pattern generator

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code) 22c. OFFICE SYMBOL none

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

19. ABSTRACT, continued

is available for testing of the AND-plane at a low cost. The use of the feedback NOR gates of the Augmented Decoder to detect stuck-at faults and short faults of the columns in the AND-plane saves a lot of hardware since another parity checker would otherwise have to be used. By using the multiplexing scheme, the sizes of the shift registers are also cut down in half compared with other PLA testing schemes.

Our work has demonstrated the advantages of using PLAs in LSI/VLSI designs. PLAs can solve the testing problem for large finite-state machines. Especially, the proposed testing scheme has all the test logic embedded on-chip. The circuits, therefore, can be tested rapidly. The test logic being built-in also greatly reduces the cost for VLSI testing since no expensive automatic test equipment will be needed. Besides, it is particularly valuable for field service because maintenance and service personnel are generally not equipped with the expensive test equipment, on which the manufacturers can base the production tests.

BUILT-IN TESTS FOR VLSI FINITE-STATE MACHINES

BY

KIEN ANH HUA

B.S., University of Illinois, 1982

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1984

Urbana, Illinois

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The introduction of Very Large Scale Integration (VLSI) technology has increased the difficulty of both designing and testing the complex systems which can be implemented on a chip. The testing problem, especially, is felt to be the primary bottleneck to the widespread use of the next generation of VLSI chips. Design for testability has been proposed as a solution to this problem [1]. Specifically, since testing of combinational circuits is much simpler than testing sequential circuits, a widely used technique for testability is Level Sensitive Scan Design [2] [3] and variants of it [4] [5]. Here, all latches in the circuit are chained together as shift registers during test, thus providing serial access to them with a very small overhead in the number of pins. Thus the latches can be tested with a set of patterns and then can be used to test the combinational logic by scanning in test patterns and scanning out results. This approach clearly does not test a circuit at speed, and the test could take a long time due to the serial nature of the scan paths.

The controllers embedded within VLSI chips are finite-state machines (FSMs). These could be broken up for testability as mentioned above. However, this Thesis would like to take a different approach to designing testable FSMs. The FSM will be implemented with a very regular structure, which is very easy to test, and the test pattern generator will be embedded on-chip for rapid testing of the FSM. The regular structure that will be used is a Programmable Logic Array (PLA) with feedback, and existing results in testing combinational PLAs [6] [7] [8] will be extended to those with feedback. The test pattern generator will be designed to take up only a relatively small amount of area in nMOS technology, the vehicle for implementing the example designs in this Thesis. This is an interesting problem since a straightforward implementation of proposed gate level designs in nMOS was found to take up excessive area.

The programmable nature of the PLA makes the design task much easier. An experiment in designing a complex system, the IBM 7441 Buffered Terminal Control Unit using PLAs is described in [9]. This paper showed that the PLA approach exploits many of the benefits of VLSI without the high engineering design costs. In addition to making circuit design easier, due to the array-oriented structure, the PLA approach for VLSI also simplifies the testing problem.

Much effort has been directed to the problem of fault detection in PLAs in recent years. Pseudo random number sequences [10] are attractive in many testing problems; this is unfortunately not an effective approach for PLA testing due to the high fan-in in PLAs [1]. If a NOR gate in the AND-plane had 20 inputs, then each random pattern would have $1/2 0^{20}$ probability of coming up with the correct input pattern. In general, only combinational logic networks with maximum fan-in of 4 can do well with random patterns. A PLA testing scheme using multiple parallel signature analyzers was proposed in [11]. This approach is suited for PLAs with a very large number of product lines compared to the number of input lines; however, typical PLAs have a ratio of number of product lines to number of input lines about four, and this approach will require a high overhead in area. The eight PLAs used in the BellMac-32A microprocessor described in [18], for instance, have this ratio ranging from 1.8 to 6.7.

A method of generating a minimal single fault detection test set from the product term specification of the PLA was presented in [12]. This testing scheme however is not appropriate for built-in tests. A universal test set for PLA was proposed in [6]. This scheme minimizes the test generation overhead. The test procedure, however, requires part of the test patterns to be generated externally and fed to the PLA inputs. This is impractical in a system environment where typical inputs to the FSM are buried within a chip and are not easily controllable.

Another built-in test approach using nonlinear feedback shift registers for test pattern generation, and multiple input signature analyzers for test response evaluation, was proposed in [7]. This approach still takes up quite a large amount of area for the test logic and for the interconnections

between the test logic and the naked PLAs, because the PLA design is very compact. The size of a typical nMOS PLA cell is 16 lambda by 16 lambda, where lambda is half the minimum feature size [13]. The test logic must be very simple, or else a large area will have to be wasted for interconnections (Figure 1(a)). This problem is particularly crucial when the number of interconnections is large.

In this Thesis, a complete built-in test design scheme for VLSI finite-state machines using PLAs will be presented. Unlike other built-in test schemes for PLA, the difficulty in interconnecting the test logic and the naked PLA will be considered. The layout has been done for an example PLA in nMOS technology to demonstrate that all the test logic cells line up perfectly with the PLA cells (Figure 1(b)). This eliminates the interconnection problem discussed previously. Also, the test logic presented in this Thesis can be cascaded to implement testable PLAs of any size. Since the proposed scheme in this Thesis conserves the regularity of the PLA structure, it is very suitable for design automation.

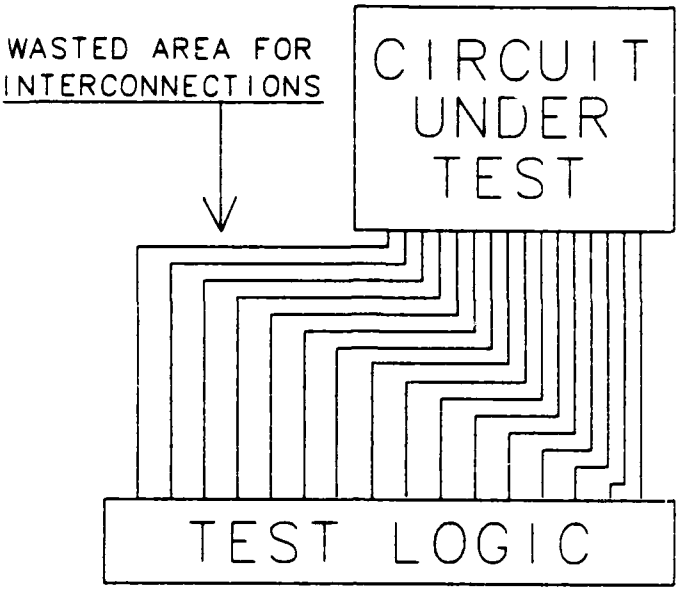


Figure 1(a) The outputs from the test logic do not line up with the inputs to the circuit under test

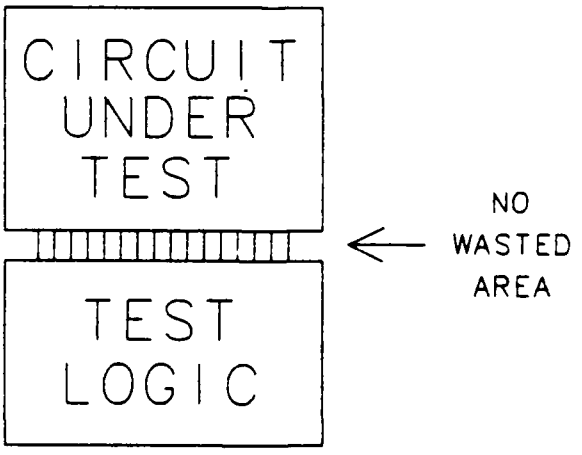


Figure 1(b) The outputs from the test logic line up with the inputs to the circuit under test

CHAPTER 2

PROGRAMMABLE LOGIC ARRAY

2.1. Design Using Programmable Logic Array

A Programmable Logic Array (PLA) is a special type of read-only memory. It consists of an AND-plane and an OR-plane as shown in Figure 2. A PLA has basically a table look-up structure, where the AND-plane forms the look-up table which has pointers pointing to the data words coded in the OR-plane. The schematic diagram for an nMOS PLA is shown in Figure 3. The inputs, stored during phase 1 in the decoders, are run vertically through the AND plane. The outputs of the AND-plane, called the product lines, leave at right angles to its inputs and run horizontally through the OR-plane. The outputs of the OR-plane then run vertically and are stored in the output register during phase 2. Notice that the planes are realized as NOR-arrays in nMOS technology. If negative logic is used, the NOR-array on the left shown in Figure 3 is functionally equivalent to an AND-plane. The NOR-array on the right together with the inverted output register forms the OR-plane.

It is clear that any combinational network can be implemented by means of sum-of-product functions as performed by PLAs. Sequential networks can also be easily realized on PLAs. As shown in Figure 4, some outputs from the OR-plane, which represent the present state of the machine, are connected to some of the inputs to the AND-plane. The value of the feedback signals is stored, along with the inputs, in the decoders during phase 1. The combined inputs then propagate through the combinational logic. The resulting outputs are stored in the output register during phase 2. Each complete machine cycle results in two new sets of outputs: the outputs of the machine and a new feedback state of the machine. The process repeats during each clock period. Thus any FSM can be easily implemented using a PLA.

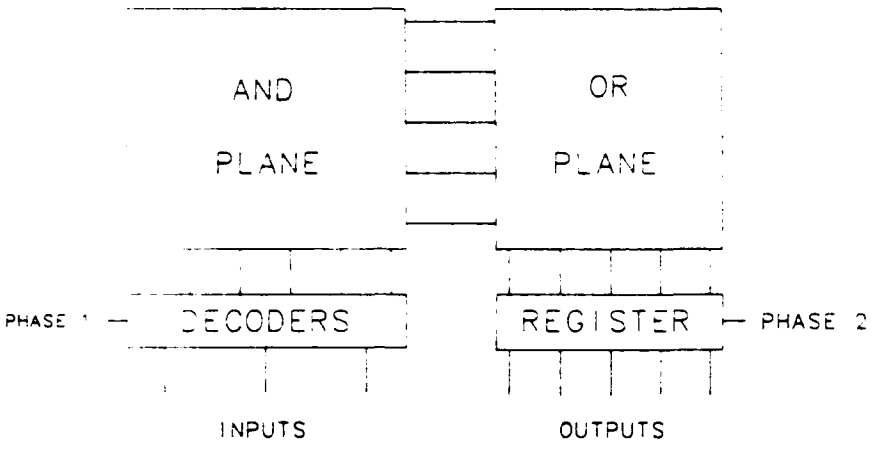


Figure 2 Overall structure of a PLA

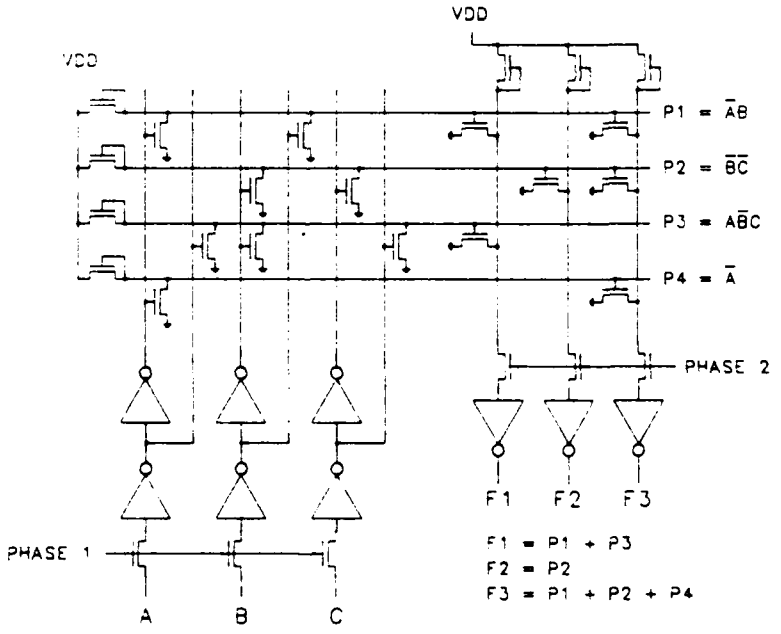


Figure 3 Circuit diagram of an nMOS PLA

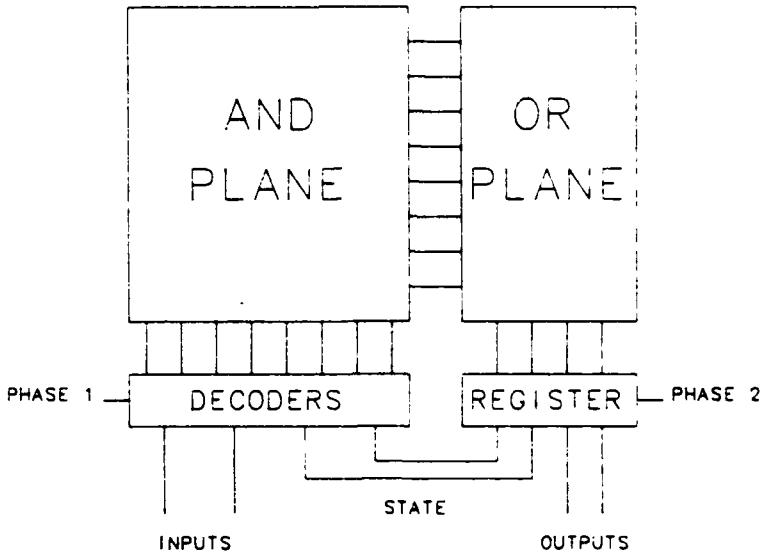


Figure 4 PLA implementation of finite-state machine

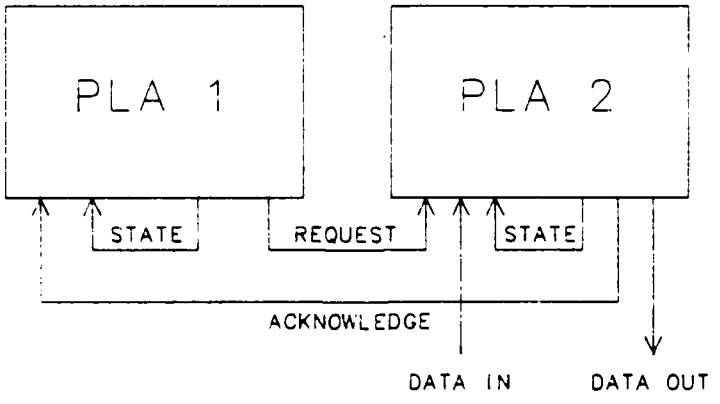


Figure 5 Finite-state machine using two PLAs

The array size of a PLA is limited because of speed considerations. Beyond some limit in the number of product terms, each column in the AND-plane becomes too long, with an increase in parasitic capacitances. The outputs of the decoders, too, have a lot of fan-out, limiting the number of product terms. Similarly, the total number of columns in the OR-plane cannot be too large. The limit in the number of inputs to a NOR array is roughly 300 today. This problem, however, can be solved by using multiple PLAs if the system to be implemented is very large. It should be noted also that PLA logic is well suited to the macro, or modular, building-block concept. A good partitioning of a system into several subsystems will simplify the design process using multiple PLAs. This is analogous to structured programming in solving the difficulty of writing complex computer programs. An example using two PLAs is shown in Figure 5. PLA1 generates a request for a unique routine in PLA2. The change of state in PLA2 is based on the external command and the internal feedback state. Words that are coded in PLA2 select the correct sequence of macros in this PLA. The end of the routine is defined by a state that generates an acknowledgement to PLA1, which in turn causes PLA1 to enter its next state.

2.2. Comparison with Random Logic Design

Owing to the array structure and programmability of PLAs, a design approach using PLAs has many advantages over a random logic approach in implementing very complex VLSI finite-state machines. Some of these are described below.

Design time is greatly reduced by the use of PLAs. Designers can use algebraic minimization which in turn can be processed by computer-aided design programs, instead of the more time-consuming design of logic gate networks. A PLA approach need not consider the placement of gates and routing of the connections. Changes in the design can also be done easily in case of mistakes or function changes. Layout is far simpler using the PLA approach; it is also less affected by working technology, design information is almost immune to changes in technology. The PLA approach also

solves the traditional problems of undetermined states ("race conditions") associated with sequential logic, the solution being provided by the clocked register-register transfer in the PLA.

Of course, although a random logic approach results in smaller chip area than a PLA approach, the most important consideration in VLSI technology is the development of very complex and error-free designs in practical amounts of time, rather than perfect chip space utilization. This is analogous to the need for high level languages in writing complex computer programs. According to [14], typical engineering manpower required for the design of a random logic integrated circuit (IC) with 10,000 transistors might be about 2 man-years. As the number of transistors in any given IC increases, the man-years of engineering design time increase exponentially. A 100,000-transistor IC composed of random logic thus could easily require a 60-man-year effort [14]. This is completely unacceptable. A design approach using regular structures will reduce the rate of increase of design time with circuit size.

The random logic approach also makes the testing of VLSI machines very difficult and time-consuming. A built-in test scheme for PLAs, described in the following chapters, will show that the testing of PLA-based logic is much less expensive.

CHAPTER 3

FINITE-STATE MACHINE WITH BUILT-IN TEST

3.1. Overall Structure of the FSM with Built-in Test

The structure of the proposed finite-state machine with Built-in Test (FSMBT) is shown in Figure 6. An extra product line and an extra output line are added to the AND and OR arrays, respectively (darker lines). The contacts on the extra product line are arranged so as to make the columns of the AND-plane all have an odd number of non-contacts. Let n be the number of output lines of the PLA. If n is odd (even), we arrange the contacts on the extra output or parity line such that each row in the OR-plane has an even (odd) number of contacts.

Two control signals T1 and T2 are used for the test. During normal operation, T1 = T2 = 0. The Test Pattern Generator (TPG) will generate the test sequence for the OR plane when T2 is set to logical 1. When T2 = 0, the TPG is isolated from the naked PLA, and is reset to its initial state. The parity checkers PC2 and PC3 (Figure 6), are used for test response evaluations when the OR-plane is under test.

Normally, T1 = T2 = 0 and the Augmented Decoder (AD) functions as a normal single bit decoder network. During the test of the OR-plane, T2 resets the AD to its initial state. When T1 is set to 1 and T2 is set to 0, the AD functions as a test pattern generator for the testing of the AND plane. The NOR gate output E1, together with the checker PC2 (Figure 6), are used for test response evaluation when the AND-plane is under test.

The previously mentioned functional modules will be described in more detail in the subsequent Sections. Table I summarizes the different operation modes of the FSMBT.

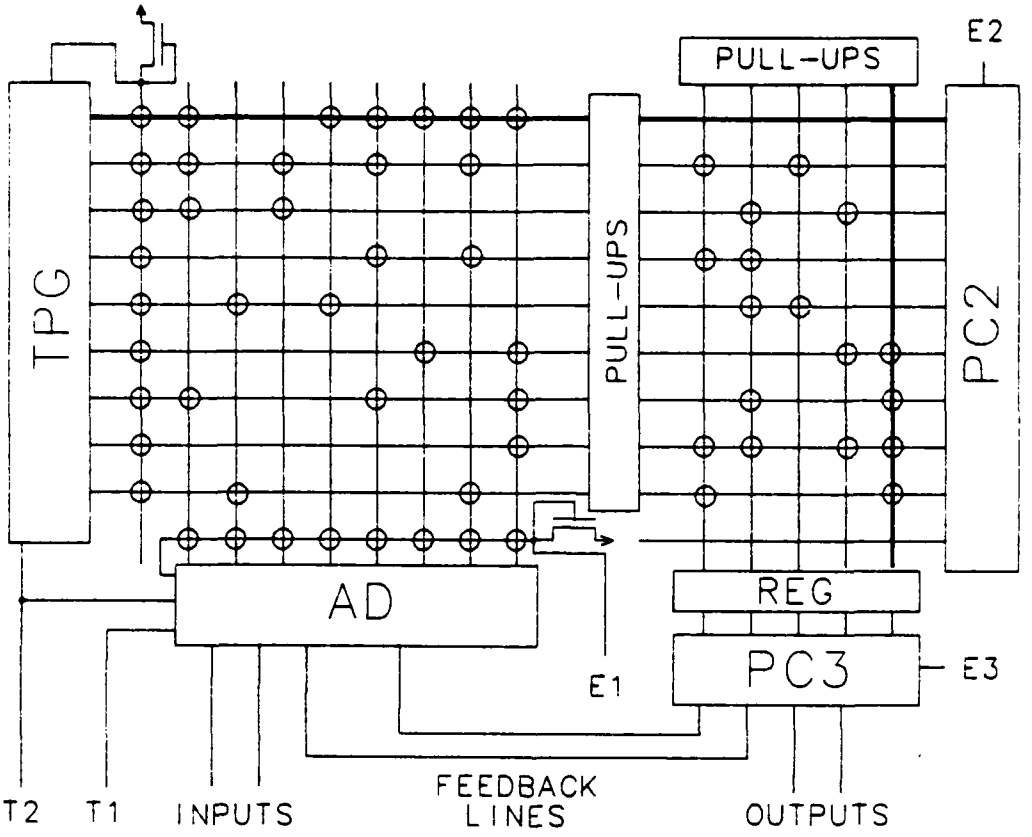


Figure 6 Structure of the FSMBT

Table I Modes of operation

T2	T1	Mode of operation
0	0	Normal machine operation
0	1	Test the AND plane
1	0	Test the OR plane
1	1	Not used

3.2. Fault Coverage

The FSMBT covers any single fault, in the naked PLA or the test logic, that is of the following types:

(1) Short between two adjacent lines. In nMOS circuits, such a short is equivalent to the wired AND of the affected lines.

(2) Stuck-at faults which cause a line to be stuck-at-1 or stuck-at-0.

(3) Contact fault due to a missing device at the cross point, or an extra device at the cross point where there should not be one.

As pointed out in [15], a great majority of physical failures are covered by shorts and stuck-at faults. The three types of faults considered in this Thesis are therefore sufficient to cover most of the physical failures in PLAs.

Although this Thesis assumes a single fault model for simplicity, most of the multiple stuck-at faults and short faults of the PLA lines (i.e., columns and rows) will also be detected using this scheme.

3.3. Test Pattern Generator

The AND-plane and the OR-plane are tested separately. The universal test set shown in this section is used for the testing of the OR-plane where F indicates a floating signal. The inclusion of the floating signals in the test set allows us to design the Test Pattern Generator (TPG) efficiently using a demultiplexing approach. This will be explained in more detail as we describe the circuit of the TPG.

The TPG that generates the test set described previously is shown in Figure 7. Initially, T2 = 0 and the TPG is isolated from the PLA; the T flip-flop and the shift register outputs are reset to logical zeros. Since all of the inputs to the NOR gate (the extra column, Figure 7) are left floating during normal PLA operation, its output shifts into the left end of the shift register a logical 1 as the the test is

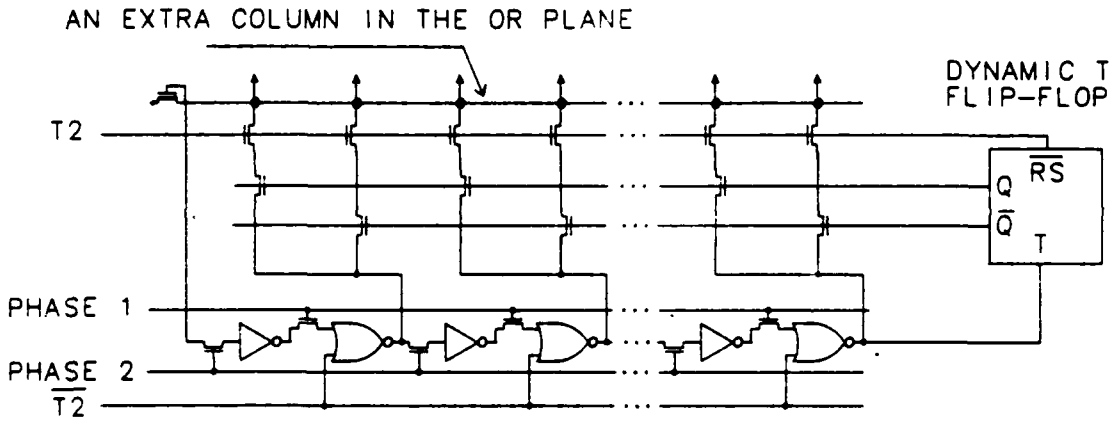


Figure 7 The Test Pattern Generator (TPG)

$$t_0 = 0F0F \_ 0F0F$$
$$t_0' = F0F0 \_ F0F0$$
$$t_1 = 1F0F \_ 0F0F$$
$$t_2 = F1F0 \_ F0F0$$
$$t_3 = 0F1F \_ 0F0F$$
$$\vdots$$
$$t_{n-1} = 0F0F \_ 0F1F$$
$$t_n = F0F0 \_ F0F1$$

turned on (i.e., T2 = 1). This 1 will subsequently be shifted along the shift register to form the test patterns with an even subscript t_2, t_4, t_6, ... As long as there is a 1 in the shift register, a zero is always shifted into its left end. The desired test patterns therefore can be obtained. When the signal 1 reaches the right end of the shift register, it is shifted out, into the T flip-flop. The flip-flop changes its state and switches the shift register outputs to the adjacent product lines which were previously left floating (demultiplexing). The test pattern t_0 and those with an odd subscript t_1, t_3, t_5 ... can then be generated.

If an n-bit shift register is to be used for an n-product term PLA, in order to have the outputs of the shift register line up with the product lines, we will have to fit at least four inverters in a width of 16 lambda. Using the design rules in [13], a pull-up generally takes up 6 lambda in width. It is therefore impossible to interconnect the four inverters within a pitch of 16 lambda to make the 2 bits of the shift register. The proposed TPG uses a shift register that is only half the size of the shift registers used in other schemes. Each shift register output is shared by two adjacent product lines by demultiplexing that can be implemented effectively using pass transistor logic. Our design not only saves on the number of transistors, but also eliminates the wasted area for interconnections of the TPG

to the AND-plane.

The checkers PC2 and PC3 can be used to detect any stuck-at fault or short fault in the TPG. This will be described next.

If any output line i of the TPG is stuck-at-0, the test t_i can detect this. If it is stuck-at-1 the test t_0 or t_0' can detect this. If any output line i of the TPG is shorted to the adjacent line i+1, the test t_i or t_{i+1} can detect this fault.

If the output of the feedback NOR gate (just an extra column in the OR-plane) is stuck-at-1 or stuck-at-0, then PC2 will detect this since a constant logic value is shifted into the TPG.

The stuck fault at the input of the T flip-flop is covered by the stuck fault at the right-most output of the shift register (Figure 7). Initially, the flip-flop is initialized to zero. If the output Q of the flip-flop is stuck-at-1, a signal 1 shifted into the shift register will be propagated to two product lines and E2 will be 0 instead of 1 under the fault-free condition (Figure 2). If output Q is stuck-at 0, eventually a 1 will be shifted into the flip-flop and both Q and $\overline{Q}$ will become zeros. Inputs to the PC2 will therefore be floating and E2 will detect this fault since E2 should be 1 whenever there is a 1 in the shift register. Testing the stuck fault at $\overline{Q}$ is similar. If Q shorts to $\overline{Q}$, the TPG is isolated from the PC2 and E2 will be carrying a constant value all the time.

During the testing of the OR-plane, T1 = 0 and T2 = 1 cause all the outputs of the AD to be preset to 0's or left floating, and E1 should be 1. If T2 is stuck-at-0, the AD functions as a decoder network. Since half of the decoder outputs will have logic 1's, E1 will become 0. This detects T2 stuck-at-0. If T2 is stuck-at-1, all the columns in the AND-plane will be floating (Figure 8). This makes E1 = 1 throughout the testing of the AND-plane. This fault therefore can be detected during the testing of the AND-plane by observing E1 (Figure 6).

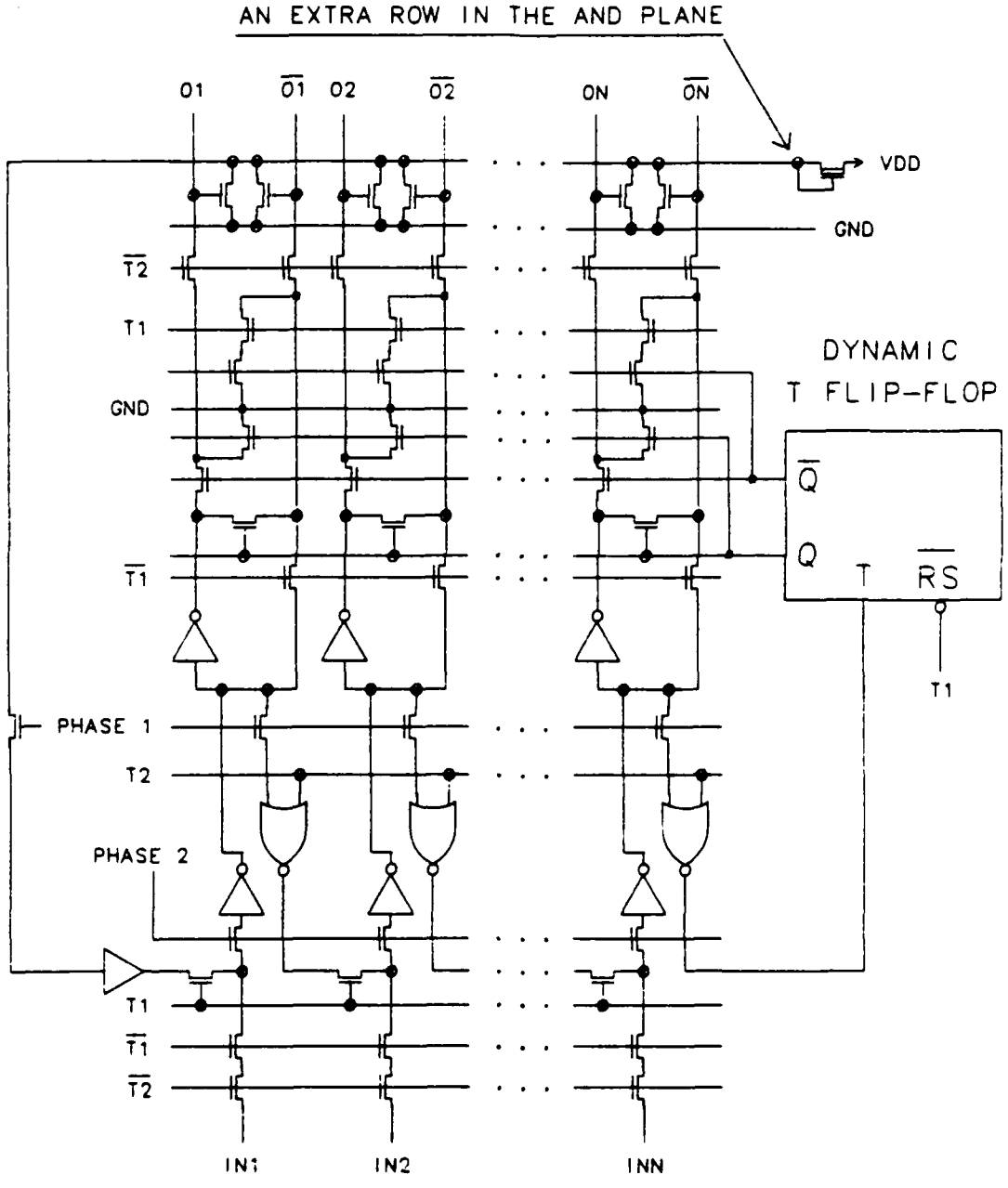


Figure 8 The Augmented Decoder (AD)

3.4. *Augmented Decoder*

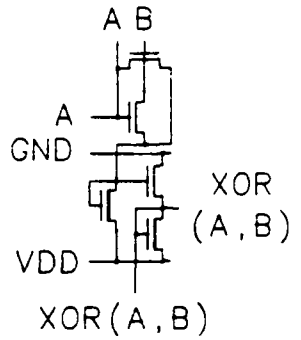
The same test set that was used for the testing of the OR-plane is used also for the testing of the AND-plane, except the floating signals are grounded. The Augmented Decoder (AD) that generates this test set is shown in detail in Figure 8. Again, it uses the multiplexing scheme described in Section 3.3. Each output of the AD is connected to its normal decoder input under normal PLA operations. When T1 becomes logical 1, it is either connected to ground or the shift register output depending on the state of the T flip-flop. The testing of the AD is similar to that of the TPG with some differences, and will be described in the following.

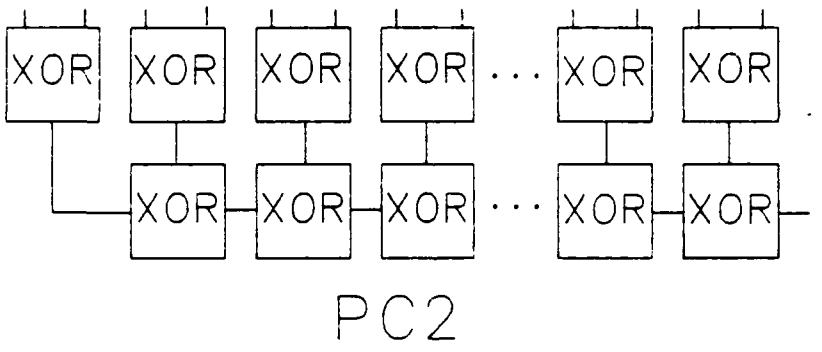
The detection of the stuck-at fault at T2 has already been described in Section 3.2. During the testing of the AND-plane, T1 = 1; if T1 is stuck-at-0, the machine is doing its normal operations, and is not in testing mode. E1 will therefore be 0 throughout the test. Since E1 = 1 is expected when the test t_0 is applied to the AND plane, the test t_0 will detect T1 stuck-at-0. During the testing of the OR-plane, E1 should be 1 throughout the test under the fault-free condition. If T1 is stuck-at-1, the AD functions as a test pattern generator even when the PLA is not in testing mode. The test pattern t_0 that makes E1 = 1 will therefore detect this fault.

Since the decoders are embedded in the shift register and the AD, testing of the shift register will also detect any fault existing in the decoders.

3.5. *Parity Checkers*

Figure 9 shows the Exclusive-OR (XOR) gate and the parity checkers implemented using the gate. Notice that two levels for the parity checkers are needed because of the compactness of the PLA cells. This will also speed up the propagation of signals along the parity checkers significantly in comparison with the scheme used in [6]. If the number of product lines is very large, it might be necessary to employ more levels to obtain the desired speed.





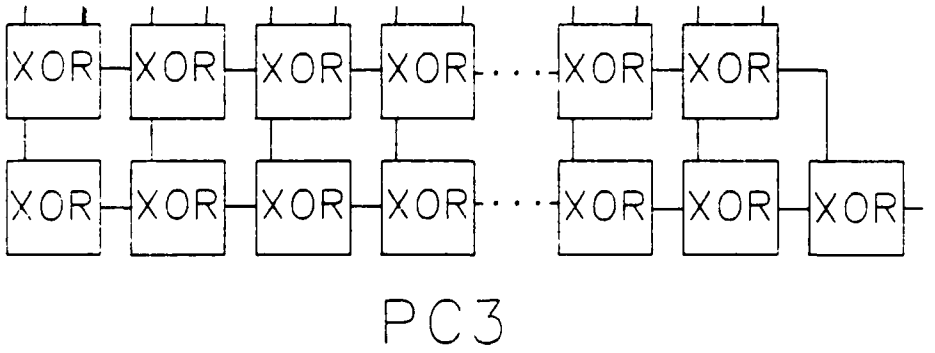


Figure 9 Parity Checker implemented using the Exclusive-OR gate

The XOR gate implemented as shown in Figure 9 needs only three vectors, (00, 01, 10), to be tested [16]. The test set generated by the TPG therefore can also detect any single fault in the PC2 circuit.

As it has been described in Section 3.1, the contacts on the extra output or parity line are arranged such that each row in the OR-plane has an even (odd) number of contacts if the total number of output lines of the PLA is odd (even). With such an arrangement, E3 can be tested for stuck-at faults. For instance, if n is odd, during the testing of the OR-plane, E3 should be 0 under the fault-free condition. This tests for E3 stuck-at-1. During the testing of the AND-plane, when the test pattern t_0 is applied, E3 should have logical 1 under the fault-free condition since there are an odd number of output lines. This tests for E3 stuck-at-0.

This Thesis assumes that no output line is redundant. An output line is redundant if it is identical to another output line, or if it outputs a constant logical value regardless of the input combinations. Such redundant outputs can always be avoided in practice. Many CAD programs are available for doing this. Although a regular PLA output line may be identical to the parity line (i.e., the extra output line), this problem can be solved by not making them adjacent. With this assumption, any stuck-at fault or short fault at the inputs of the PC3 can be detected during the testing of the OR-plane. An output line i can be detected for stuck-at-1 (or stuck-at-0) when it is supposed to have value 0 (or 1). If an output line i shorts to the adjacent line i+1, this fault will be detected when the two lines carry opposite values.

If the detection of the faults only at the external lines of PC3 is not sufficient, a technique reported in [8] can be applied. Only four test patterns are necessary to test the entire parity tree composed of 2-input XOR gates. By adding four extra product lines, this small test set can be generated efficiently. This test set can also detect any cross point fault on the extra product lines [8]. Alternatively, we can use a 2-rail parity checker [16] in place of the PC3. A 2-rail parity checker is a totally self-checking (TSC) circuit. Loosely speaking, a circuit is TSC if a fault in the circuit cannot cause an

error in the outputs without detection of the fault. The 2-rail parity checker can detect any error at the input code word, as well as any faults that may occur in the checker itself, as long as they do not occur at the same time. The cell we designed for implementing the PC3 can also be used to design the 2-rail parity checker. It takes up slightly less area than the PC3. An extra pin however is required.

3.6. Testing of the Output Register and the NOR-Planes

Stuck-at faults and short faults at the product lines are covered by stuck-at faults and short faults of the inputs to PC2 respectively. A stuck-at fault or short fault at the output lines of the PLA is covered by the stuck-at fault or short fault of the inputs to PC3 respectively. Stuck-at faults and short faults of the bit lines (i.e., output lines from the decoders) are covered by stuck-at faults and short faults of the outputs of the AD respectively. The tests of the modules PC2, PC3 and AD also test all the lines in the PLA planes.

If a missing contact or an extra contact fault occurs on the column i of the AND-plane, PC2 will detect this fault during the testing of the AND-plane when the test pattern t_i is applied. Similarly, a contact fault on the rows of the OR-plane can be detected by PC3 during the testing of the OR-plane.

3.7. Test Response Evaluation

Let m/2 be the number of inputs, p be the number of product lines and n be the number of output lines. Table II summarizes the test responses under the fault-free condition. The total number of test patterns is m + p + 2.

In the conventional testing scheme, test responses are evaluated by comparing them to the previously obtained correct responses one by one (Figure 10). This scheme is clearly not suitable for built-in tests. An alternative is to use a T flip-flop as a serial input parity checker that compresses the test response. This approach allows the evaluation of the test to be done only at the end of the test. For instance, the test response sequence being inputed serially into the T flip-flop is <11010001>. If the

Table II Test responses under fault free conditions

Test mode T2	Test mode T1	Product lines P_1	P_2	P_3	...	P_{p-1}	P_p	Bit lines B_1	B_2	B_3	...	B_{m-1}	B_m	Correct response E1	E2	E3
0	1	1	1	1	...	1	1	0	0	0	...	0	0	1	g	-
0	1	1	1	1	...	1	1	0	0	0	...	0	0	1	g	-
0	1	1	1	1	...	1	1	1	0	0	...	0	0	0	1	-
0	1	1	1	1	...	1	1	0	1	0	...	0	0	0	1	-
.	.			.						.				.	.	.
.	.			.						.				.	.	.
.	.			.						.				.	.	.
0	1	1	1	1	...	1	1	0	0	0	...	1	0	0	1	-
0	1	1	1	1	...	1	1	0	0	0	...	0	1	0	1	-
1	0	0	F	0	...	0	F	F	F	F	...	F	F	1	0	0
1	0	F	0	F	...	F	0	F	F	F	...	F	F	1	0	0
1	0	1	F	0	...	0	F	F	F	F	...	F	F	1	1	e
1	0	F	1	F	...	F	0	F	F	F	...	F	F	1	1	e
.	.			.						.				.	.	.
.	.			.						.				.	.	.
.	.			.						.				.	.	.
1	0	0	F	0	...	1	F	F	F	F	...	F	F	1	1	e
1	0	F	0	F	...	F	1	F	F	F	...	F	F	1	1	e

e = 1 if n is even
e = 0 if n is odd
g = 1 if p is odd

- is don't care
F denotes a floating signal
g = 0 if p is even

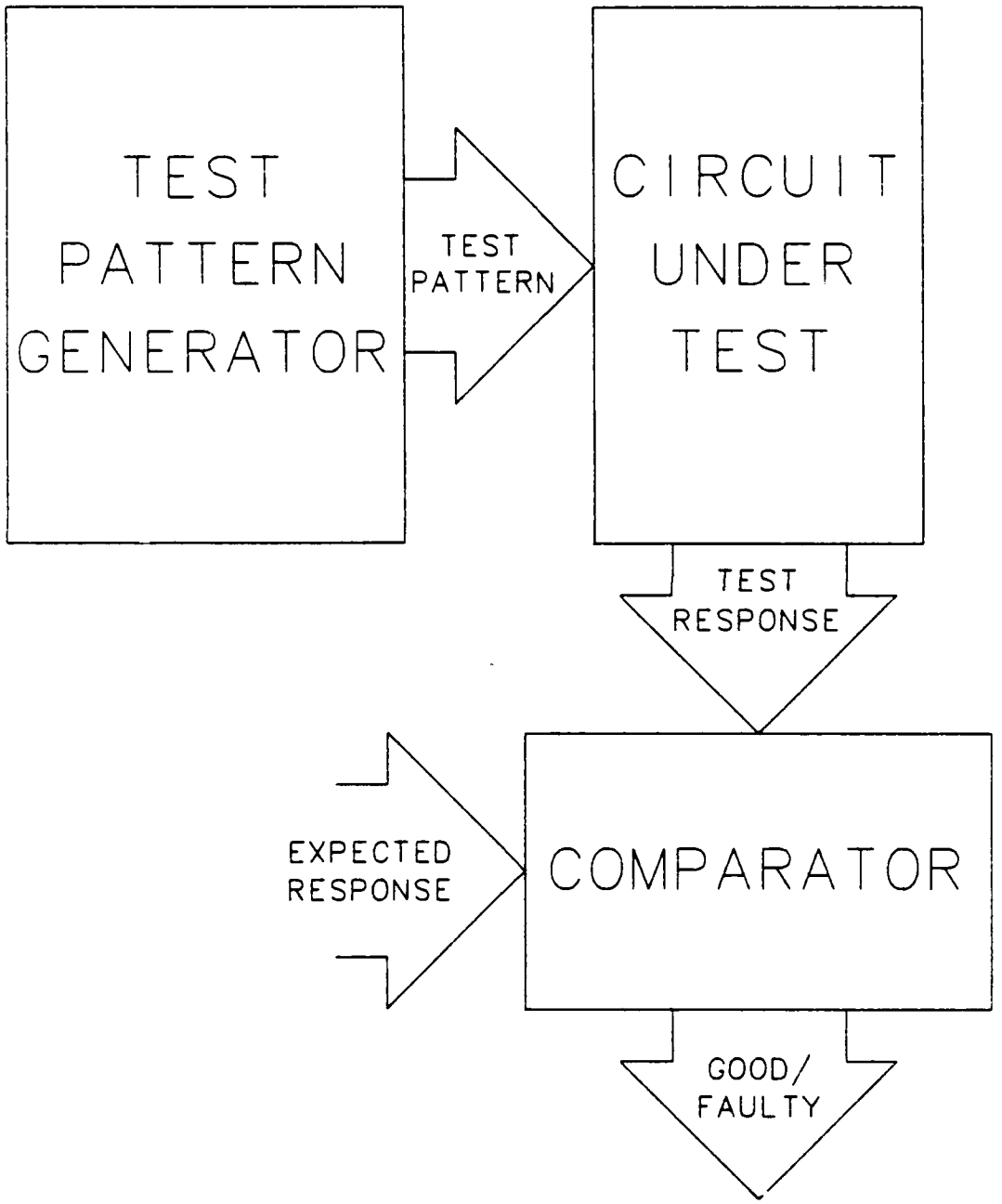


Figure 10 Conventional test response evaluation scheme

flip-flop is initially reset to 0, the expected final output from the flip-flop at the end of the input sequence would be 0 because there are an even number of 1's in the input sequence. Using these T flip-flops as a second-level test-response evaluator to compress the test response sequences from E1, E2 and E3, any single fault detected by E1, E2 or E3 will change the parity of the serial input sequences to the corresponding flip-flops, and thus their final outputs.

CHAPTER 4

OVERHEAD AND COMPARISON WITH OTHER SCHEMES

Layouts for the test latch cells have been done in order to study the actual overhead of the proposed built-in test scheme. A layout example with 32 inputs, 18 feedback paths, 190 product terms and 49 outputs as shown in Figure 11 was studied. This PLA is identical in size to one of the eight PLAs used in a PLA design for the BellMac-32A microprocessor described in [18]. The overhead for this example is 20%.

A built-in test example for PLA with 10 inputs, 30 outputs and 150 product terms was presented in [11]. The overhead was estimated in [11] to be 23%. This example has a ratio of 15 for the number of product terms to number of inputs. The overhead will become worse when this ratio is lower as it is usually the case. The eight PLAs in the BellMac microprocessor [18], for instance, have this ratio ranging from 1.8 to 6.7. This scheme [11] also becomes impractical when the number of inputs is quite large, because 2^n test patterns would be required where n is the number of outputs from the decoders. In the example above (of a PLA similar to that used in the BellMac-32A [18]), the number of test patterns required by [11] would be 10^{15}.

The testing scheme presented in [7] uses the built-in logic block observer (BILBO) scheme [5]. Since BILBO takes up even more area than the linear feedback shift register used in [11], and the number of product terms is usually more than double the number of inputs, the overhead for the scheme presented in [7] is generally worse than that for the scheme presented in [11]. For the example used above, namely a PLA with 32 inputs, 18 feedbacks, 190 product terms and 49 outputs, the overhead is estimated to be 50% if the scheme presented in [7] is used.

The scheme proposed in this Thesis is thus good for VLSI finite-state machines when large digital systems are implemented using PLAs such as the one used in the BellMac microprocessor [18]. Figure

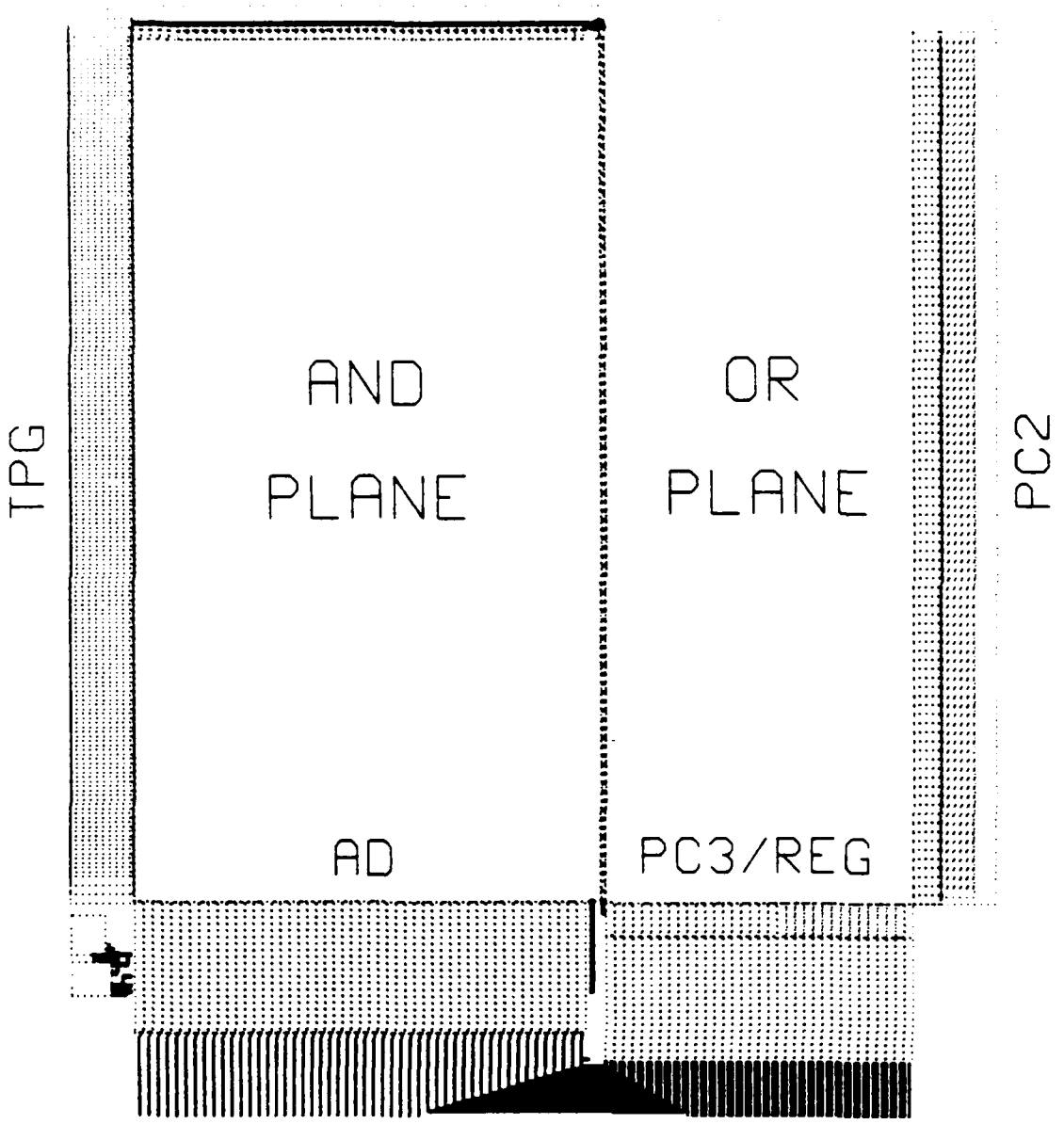


Figure 11 A layout example, the overhead is 20%

12 shows the overhead estimates for a wide range of PLA parameters, where p is the number of product terms and N is the sum of the number of inputs (m), the number of outputs (n), and the number of feedback lines (f). It was assumed that m = n = f in obtaining these graphs. Notice that the overhead becomes smaller as the size of the PLA increases.

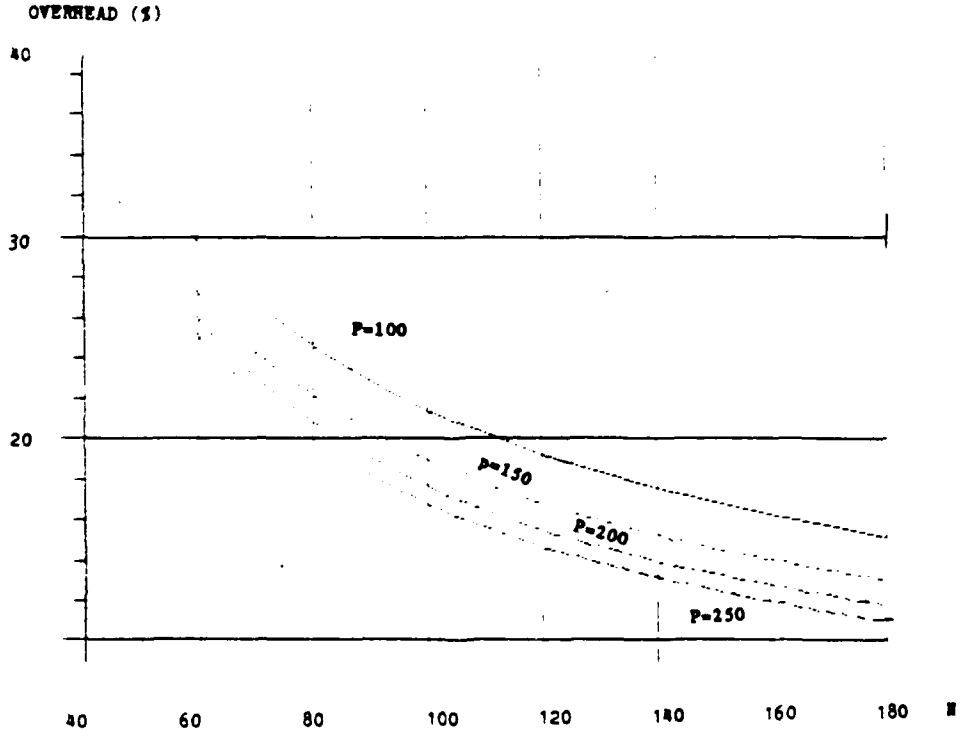


Figure 12 Overhead estimates

CHAPTER 5

CONCLUSIONS

Using a universal test set allows a finite-state machine implemented with a PLA to be tested within a number of cycles that is proportional to the number of inputs and product terms. By modifying the single-bit decoders, a test pattern generator is available for the testing of the AND-plane at a low cost. The use of the feedback NOR gates of the Augmented Decoder to detect stuck-at faults and short faults of the columns in the AND-plane saves a lot of hardware since another parity checker would otherwise have to be used. By using the demultiplexing scheme, the sizes of the shift registers are also cut down in half compared with other PLA testing schemes. If the floating signals in the test set are grounded, this demultiplexing scheme can also be used in the design of the parity checkers to further cut down the overhead. In this thesis, we used 2x1 demultiplexing networks. This can be extended to employ 4x1 or even 8x1 demultiplexing networks for those very large PLAs. This way, a large PLA is partitioned into smaller PLAs during the test so that they can share the built-in test hardware. Furthermore, the proposed built-in test scheme for PLAs is applicable to high density folded PLAs. This makes the design approach using PLAs even more attractive for VLSI technology.

The scheme proposed in this Thesis has thus achieved a FSMBT with very low overhead. Typical overhead ratios will be about 20%, which is reasonably small for VLSI chips. The test logic cells can be cascaded to accommodate PLAs of any desired size since the proposed scheme matches the regularity of the PLA structure. Existing PLA synthesis programs can be modified for designing such FSMBTs.

Papers such as [9] and [18] have demonstrated the advantages of using PLAs in LSI/VLSI designs. A good approach for VLSI requires also an efficient testing strategy. This Thesis shows how PLAs can solve the testing problem for large finite-state machines. Especially, the proposed testing scheme has

all the test logic embedded on-chip. The circuits therefore can be tested rapidly. The test logic being built-in also greatly reduces the cost for VLSI testing since no expensive automatic test equipment will be needed. Besides, it is particularly valuable for field service because maintenance and service personnel are generally not equipped with the expensive test equipments, on which the manufacturers can base the production tests.

APPENDICES

TEST LOGIC CELLS

The test logic presented in this Thesis can be cascaded to implement testable PLAs of any size. Since the proposed scheme in this Thesis conserves the regularity of the PLA structure, it is very suitable for design automation.

In the subsequent pages, the layouts, logic circuit diagrams and descriptions of the test logic cells will be presented. The circuits for the T flip-flop will not be shown since its dimensions are not critical. Almost any existing T flip-flop with reset capability can be used for the proposed scheme.

APPENDIX A

TPGCELL

NAME

TPGCELL - Test Pattern Generator cell (Figure A.1)

DESCRIPTION

This cell is used in the implementation of the test pattern generator for the testing of the OR-plane. It is isolated from the PLA when T2 = 0. When T2 = 1, it generates the test patterns (Section 3.3).

DIMENSION

112 Lambda x 16 Lambda

INTERFACE

T2	input,	test control signal
Q	input,	output of the T flip-flop
VDD	input,	power supply
GND	input,	ground
PHASE 1	clock,	switch control
PHASE 2	clock,	switch control
SHF-IN	input,	shift-in
SHF-OUT	output,	shift-out
O1	output,	output to OR-plane
O2	output,	output to OR-plane

TPGCELL

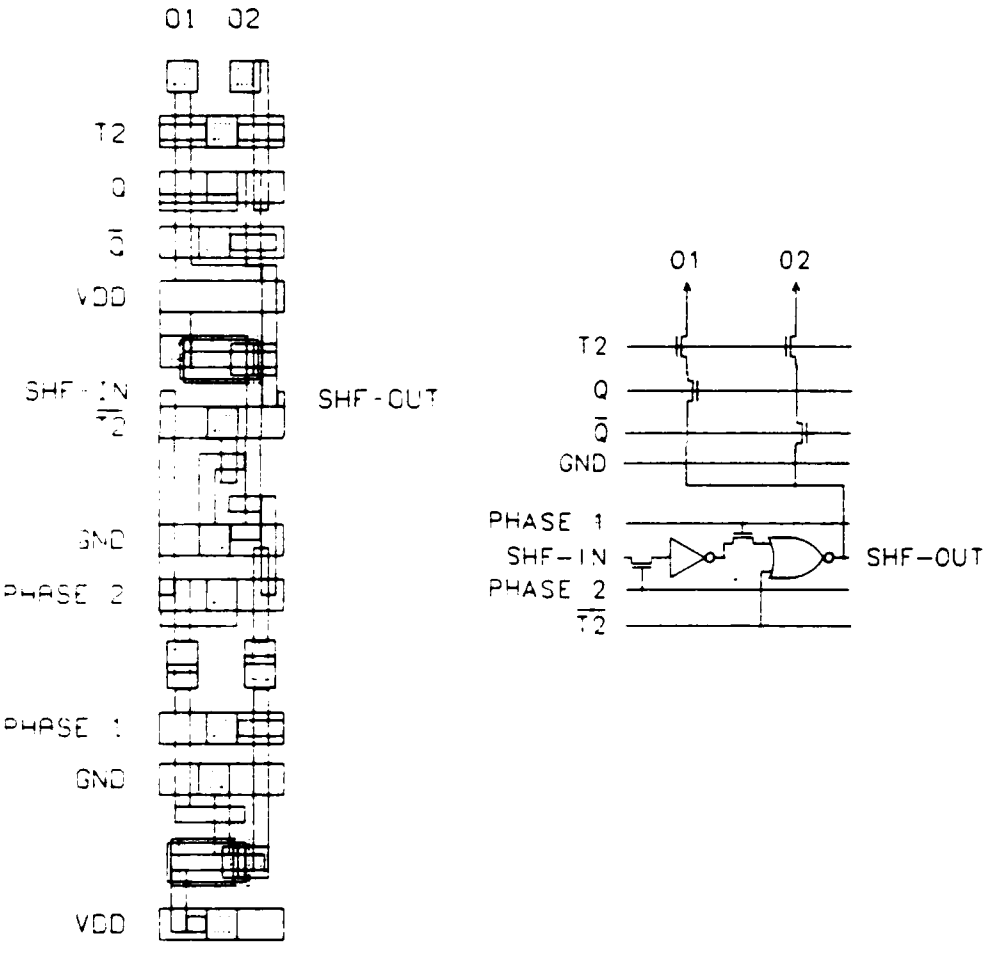


Figure A.1 TPGCELL cell

APPENDIX B

ADCELL

NAME

ADCELL - Augmented Decoder Cell (Figure A.2)

DESCRIPTION

This cell is used in the implementation of the augmented decoder. It behaves like a clocked PLA decoder under normal PLA operation (i.e., T1 = T2 = 0). When T1 = 1 and T2 = 0, it becomes a test pattern generator for the testing of the AND-plane.

DIMENSION

232 Lambda x 16 Lambda

INTERFACE

T1	input,	test control signal
T2	input,	test control signal
Q	input,	output of the T flip-flop
VDD	input,	power supply
GND	input,	ground
OUT	output,	true output to AND-plane
$\overline{\text{OUT}}$	output,	complement output to AND-plane
SHF-IN	input,	shift-in
SHF-OUT	output,	shift-out
PHASE 1	clock,	switch control
PHASE 2	clock,	switch control
IN	input,	decoder input

ADCELL

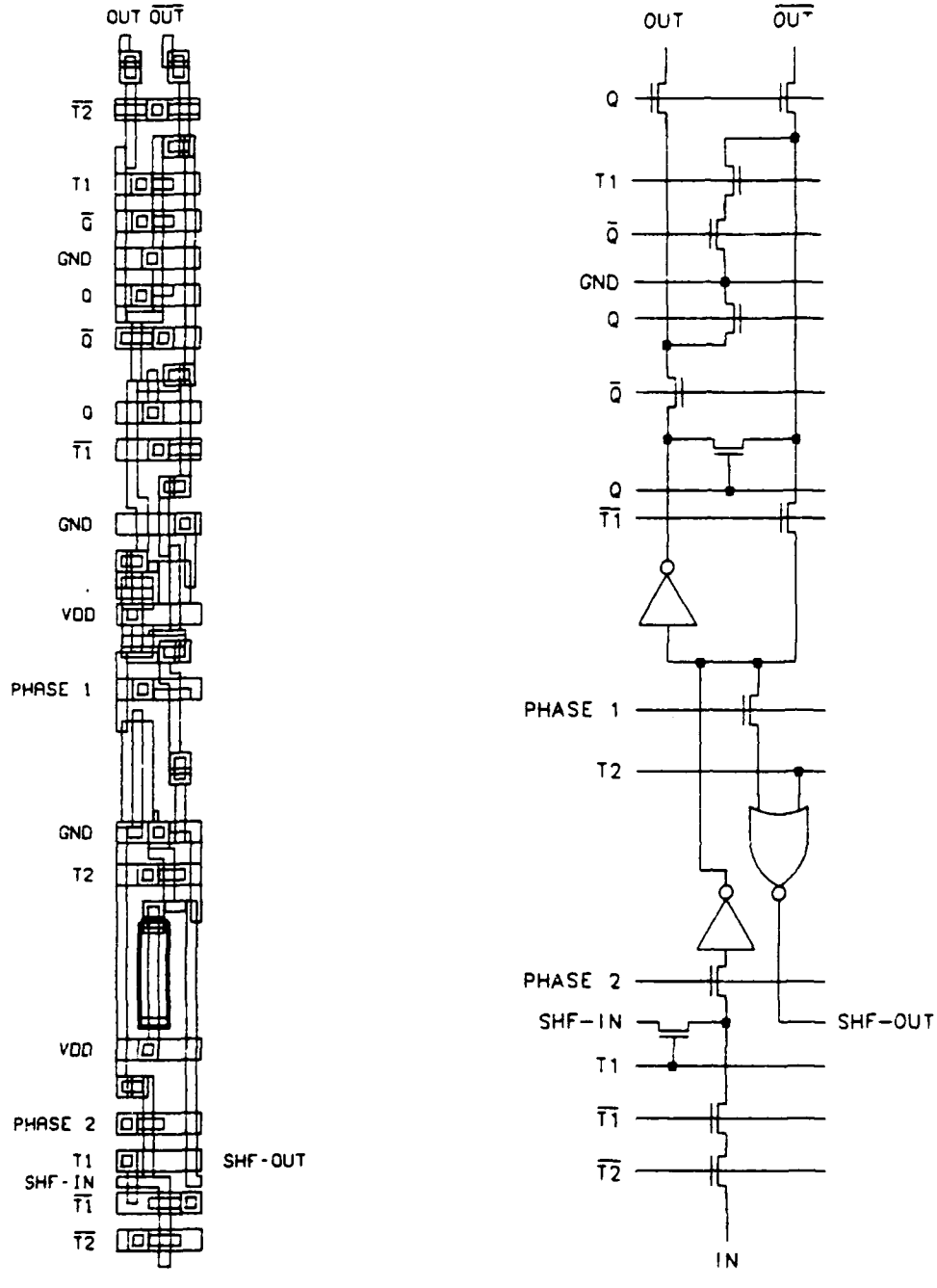


Figure A.2 ADCELL cell

APPENDIX C

XOR1

NAME

XOR1 - Exclusive OR gate (Figure A.3)

DESCRIPTION

This cell functions as a 2-input Exclusive-OR gate. It is used to implement the first or upper level of the parity checker PC2.

DIMENSION

50 Lambda x 16 Lambda

INTERFACE

IN1	input,	
IN2	input,	
OUT	output,	XOR (IN1,IN2)
VDD	input,	power supply
GND	input,	ground

XOR1

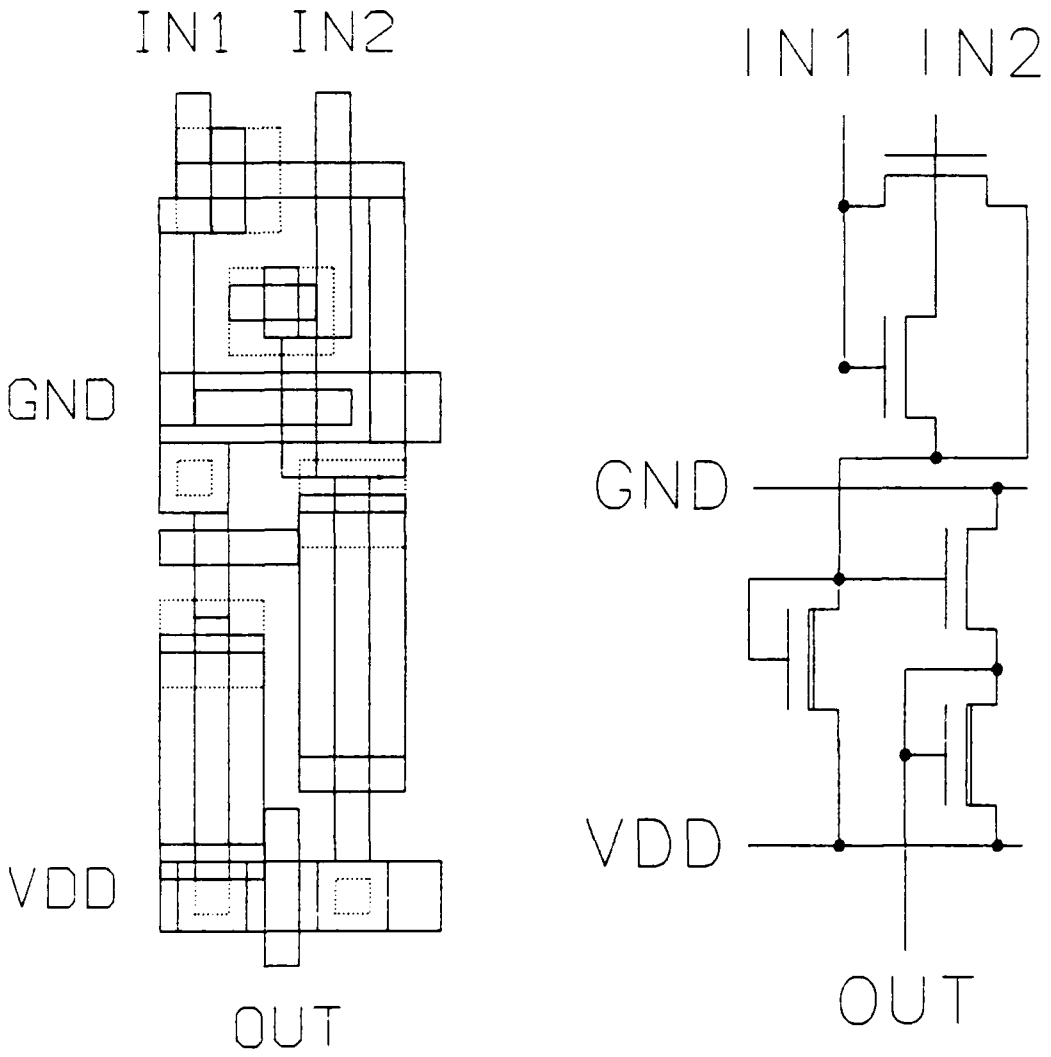


Figure A.3 XOR1 cell

APPENDIX D

XOR2

NAME

XOR2 - Exclusive OR gate (Figure A.4)

DESCRIPTION

This cell functions as a 2-input Exclusive-OR gate. It is used to implement the second or lower level of the parity checker PC2.

DIMENSION

58 Lambda x 16 Lambda

INTERFACE

IN1	input,	
IN2	input,	
OUT	output,	XOR (IN1,IN2)
VDD	input,	power supply
GND	input,	ground

XOR2

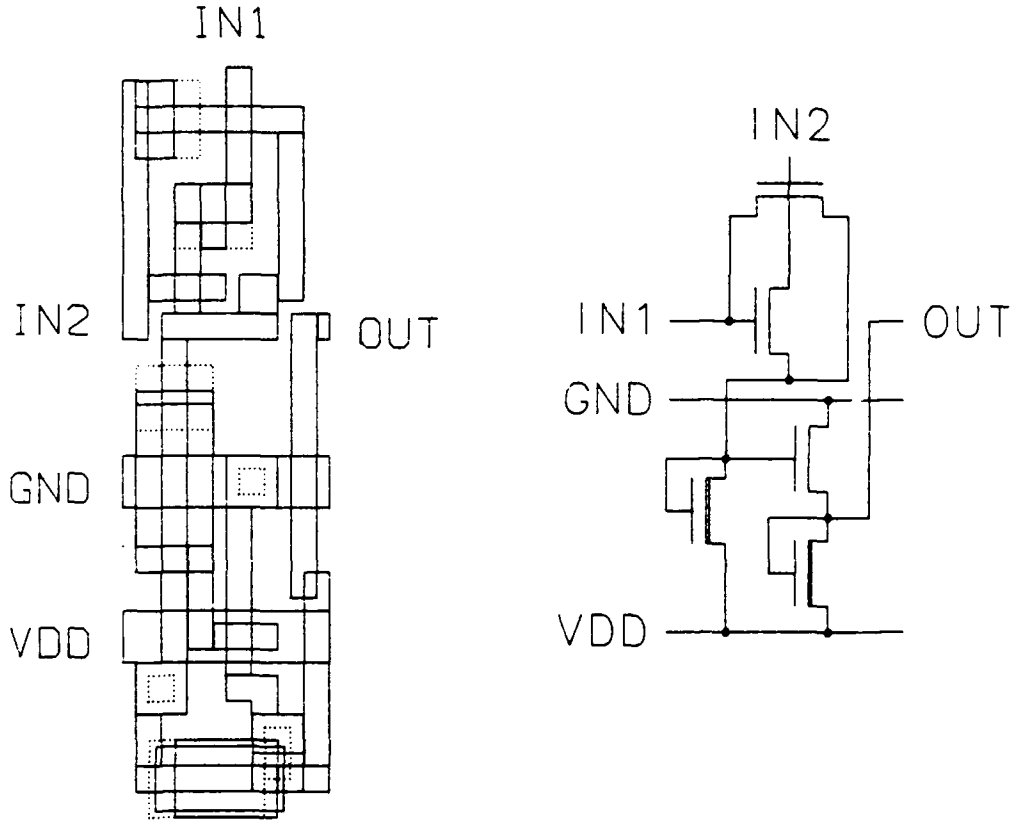


Figure A.4 XOR2 cell

APPENDIX E

XOR3

NAME

XOR3 - Exclusive OR gates (Figure A.5)

DESCRIPTION

This cell functions as two 2-input Exclusive-OR gates. It is used to implement the parity checker PC3. The two outputs can be exclusive-ORed to form a single final output or they can be used as the two outputs of a totally self-checking parity checker [16].

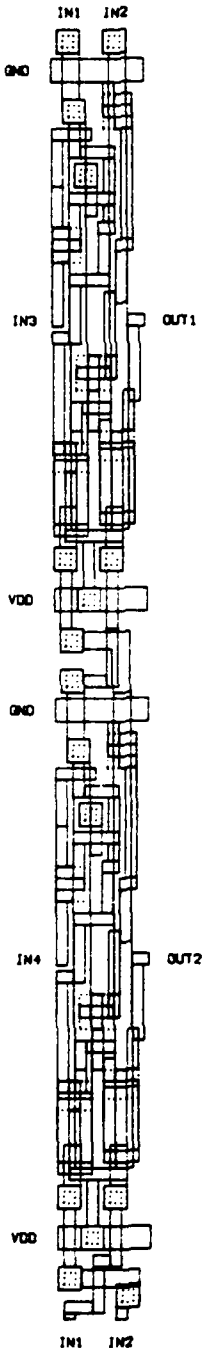
DIMENSION

222 Lambda x 16 Lambda

INTERFACE

IN1	input,	
IN2	input,	
IN3	input,	
IN4	input,	
OUT1	output,	XOR (IN1,IN3)
OUT2	output,	XOR (IN2,IN4)
VDD	input,	power supply
GND	input,	ground

XOR3



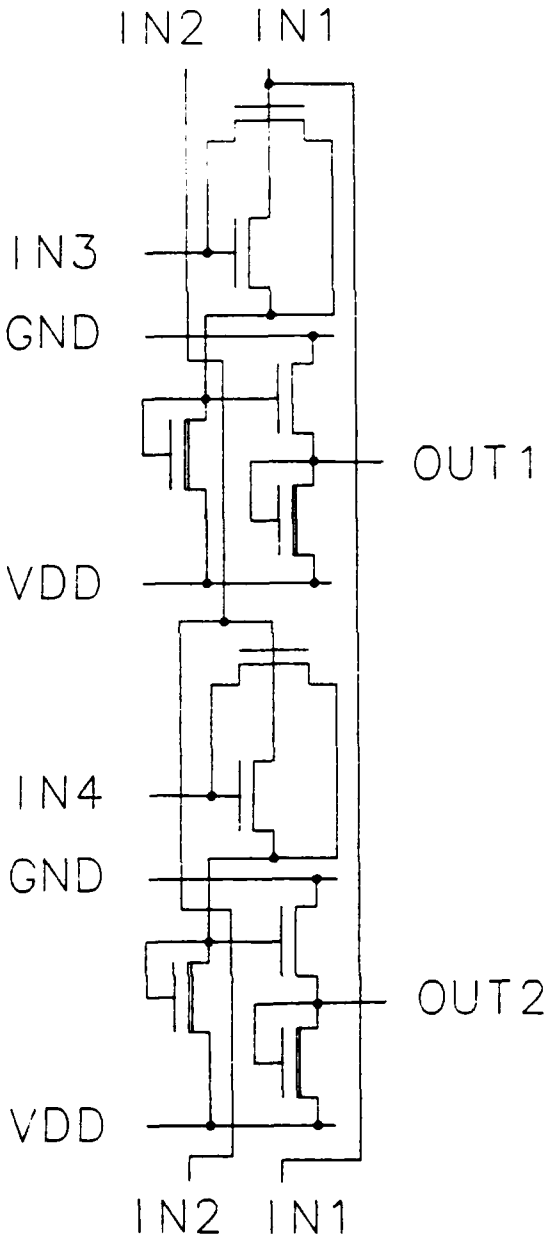


Figure A.5 XOR3 cell

REFERENCES

[1] T. W. Williams and K. P. Parker, "Design for Testability - A Survey," *IEEE Trans. Comput.*, Vol. c-31, No. 1, pp. 2-15, Jan. 1982.

[2] M. J. Williams and J. B. Angell, "Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic," *IEEE Trans. Comput.*, Vol. C-22, pp. 46-60, Jan. 1973.

[3] E. B. Eichelberger and T. W. Williams, "A Logic Design Structure for LSI Testability," *Proc. 14th Design Automation Conference*, pp. 462-468, June 1977.

[4] J. H. Stewart, "Future Testing of Large LSI Circuit Cards," *Dig. 1977 Semiconductor Test Symp.*, pp. 6-17, Oct. 1977.

[5] B. Koenemann, J. Mucha and G. Zwiehoff, "Built-in Logic Block Observation Techniques," *Dig. 1979 Test Conf.*, pp. 37-41, Oct. 1979.

[6] H. Fujiwara and K. Kinoshita, "A Design of Programmable Logic Arrays with Universal Tests," *IEEE Trans. Comput.*, Vol. C-30, No. 11, pp. 823-828, Nov. 1981.

[7] W. Daehn and J. Mucha, "A Hardware Approach to Self-testing of Large Programmable Logic Arrays," *IEEE Trans. Comput.*, Vol. C-30, No. 11, pp. 829-833, Nov. 1981.

[8] S. J. Hong and D. L. Ostapko, "FITPLA: A Programmable Logic Array for Function Independent Testing," *Dig. 10th Int'l. Symp. on Fault-Tolerant Computing*, pp. 131-136, Oct. 1980.

[9] J. C. Logue, N. F. Brickman, F. Howley, J. W. Jones and W. W. Wu, "Hardware Implementation of A Small System in Programmable Logic Arrays," *IBM J. Res. Develop.*, pp. 110-119, Mar. 1975.

[10] D. K. Bhavsar and R. W. Heckelman, "Self-testing by Polynomial Division," *Proc. IEEE Int'l. Test Conf.*, pp. 208-216, 1981.

[11] S. Z. Hassan and E. J. McCluskey, "Testing PLAs Using Multiple Parallel Signature Analyzers," *Proc. 13th Int'l. Symp. on Fault-Tolerant Computing*, pp. 422-425, June 1983.

[12] P. Bose and J. A. Abraham, "Test Generation for Programmable Logic Arrays," *IEEE 19th Design Automation Conf.*, pp. 574-580, 1982.

[13] C. Mead and L Conway, *Introduction to VLSI Systems. Reading, Massachusetts:* Addison-Wesley, 1980.

[14] Kent F. Smith, Tony M. Carter and Charles E. Hunt, "Structure Logic Design of Integrated Circuits Using the Storage/Logic Array (SLA)," *IEEE J. of Solid-State Circuits*, Vol. SC-17, No. 2, pp. 395-406, April 1982.

[15] J. Galiay, Y. Crouzet and M. Vergniault, "Physical Versus Logical Fault Models MOS LSI Circuits: Impact on Their Testability," *IEEE Trans. Comput.*, Vol. c-29, No. 6, pp. 527-531, June 1980.

[16] Y. Crouzet, "The P. A. D.: A Self-Checking LSI Circuit for Fault-Detection in Microcomputers," *Proc. 12th Int'l. Symp. on Fault-Tolerant Computing*, pp. 55-62, June 1982.

[17] D. K. Pradhan and J. J. Stiffer, "Error-Correcting Codes and Self-Checking Circuits," *IEEE Computer*, pp. 27-37, Mar. 1980.

[18] H-F. S. Law and M. S. Shoji, "PLA Design for the BellMac-32A Microprocessor," *Proc. 1982 Int'l. Conf. on Circ. and Comput.*, pp. 161-164, Sept. 1982.

END

FILMED

2-85

DTIC